# SPRING INJECTING INNER BEANS

As you know Java inner classes are defined within the scope of other classes, similarly, **inner beans** are beans that are defined within the scope of another bean. Thus, a <bean/> element inside the <property/> or <constructor-arg/> elements is called inner bean and it is shown below.

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

   <bean  >
      <property name="target">
         <bean  />
      </property>
   </bean>

</beans>
```

## Example:

Let us have working Eclipse IDE in place and follow the following steps to create a Spring application:

| Step | Description |
|---|---|
| 1 | Create a project with a name *SpringExample* and create a package *com.tutorialspoint* under the **src** folder in the created project. |
| 2 | Add required Spring libraries using *Add External JARs* option as explained in the *Spring Hello World Example* chapter. |
| 3 | Create Java classes *TextEditor*, *SpellChecker* and *MainApp* under the *com.tutorialspoint* package. |
| 4 | Create Beans configuration file *Beans.xml* under the **src** folder. |
| 5 | The final step is to create the content of all the Java files and Bean Configuration file and run the application as explained below. |

Here is the content of **TextEditor.java** file:

```
package com.tutorialspoint;

public class TextEditor {
   private SpellChecker spellChecker;

   // a setter method to inject the dependency.
   public void setSpellChecker(SpellChecker spellChecker) {
      System.out.println("Inside setSpellChecker." );
      this.spellChecker = spellChecker;
   }
   // a getter method to return spellChecker
   public SpellChecker getSpellChecker() {
      return spellChecker;
   }
   public void spellCheck() {
      spellChecker.checkSpelling();
   }
}
```

Following is the content of another dependent class file **SpellChecker.java**:

```
package com.tutorialspoint;

public class SpellChecker {
   public SpellChecker(){
      System.out.println("Inside SpellChecker constructor." );
   }

   public void checkSpelling(){
      System.out.println("Inside checkSpelling." );
   }

}
```

Following is the content of the **MainApp.java** file:

```
package com.tutorialspoint;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

public class MainApp {
   public static void main(String[] args) {
      ApplicationContext context =
             new ClassPathXmlApplicationContext("Beans.xml");

      TextEditor te = (TextEditor) context.getBean("textEditor");

      te.spellCheck();
   }
}
```

Following is the configuration file **Beans.xml** which has configuration for the setter-based injection but using **inner beans**:

```
<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

   <!-- Definition for textEditor bean using inner bean -->
   <bean  >
      <property name="spellChecker">
         <bean  />
       </property>
   </bean>

</beans>
```

Once you are done with creating source and bean configuration files, let us run the application. If everything is fine with your application, this will print the following message:

```
Inside SpellChecker constructor.
Inside setSpellChecker.
Inside checkSpelling.
```